

Curiosity Cloning Image Viewer User's Manual

Marek Ruciński
Advanced Concepts Team, European Space Agency
marek.rucinski@esa.int

ACT Technical Report, ACT-MAN-5100-CCIVUM01
November 18, 2008

Contents

1	Introduction	1
1.1	Features	1
1.2	Software Requirements	2
1.3	Legal Information	2
2	Principles of Operation	5
2.1	Terminology	5
2.2	Program Operation	5
3	Configuration Dialog	7
3.1	Group “Display”	8
3.1.1	“Display adapter”	8
3.1.2	“Display mode”	8
3.1.3	“Fullscreen”	8
3.2	Group “Datasets”	8
3.2.1	“Image folder”	8
3.2.2	“Use random image sequence(s)” / “Use an experiment definition file”	8
3.2.3	“Number of images per run”	9
3.2.4	“Random seeds”	9
3.2.5	Experiment definition file (not labelled)	9
3.3	Group “Exposition”	9
3.3.1	“Attempt to achieve arbitrary durations” / “Strictly synchronize with monitor’s vertical refresh”	9
3.3.2	“IDP duration”	9
3.3.3	“IIP duration”	9
3.3.4	“Synchronize with monitor’s vertical refresh”	9
3.3.5	“Allow shorter IDP durations”	10
3.3.6	“Allow shorter IIP durations”	10
3.3.7	“IDP multiplier”	10
3.3.8	“IIP multiplier”	10
3.3.9	“Show fixation cross over images”	10
3.3.10	“Fixation / rest duration”	10

3.4	Group “Performance”	10
3.4.1	“Frame dimensions”	11
3.4.2	“Image buffer size”	11
4	Usage Guidelines	12
4.1	Video Mode Selection	12
4.2	Setting-up the Experiment	13
4.3	Timing Options	14
4.3.1	Timing Basics	15
4.3.2	Arbitrary Timing	16
4.3.3	Synchronised Timing	20
4.3.4	Final Remarks	22
4.4	Performance Considerations	22
4.5	Logging	24

List of Figures

3.1	CCViewer configuration dialog	7
4.1	Frame rendering algorithm	15
4.2	The basic timing mechanism	16
4.3	Outcome of the arbitrary timing approach	17
4.4	Due-time relative arbitrary timing approach	17
4.5	Timing histograms for arbitrary timing	18
4.6	Outcome of the arbitrary timing method with vertical refresh synchronisation	19
4.7	Timing histograms for synchronised timing	20
4.8	Outcome of the synchronised timing method	21

List of Tables

4.1	Exposition times for popular refresh rates	21
-----	--	----

Chapter 1

Introduction

Curiosity Cloning Image Viewer (further referred to as CCViewer) is an application which has been designed to display images with very high timing accuracy. The application has been initially developed by the Advanced Concepts Team, European Space Agency in order to provide a tool to be used by the universities participating in the “Curiosity Cloning – Neural Modelling for Image Analysis” project¹. The software is intended to be used to display series of images to a human subject, during which it’s EEG signals and possibly other biometric measurements are taken.

1.1 Features

The software allows the user to:

- Choose the display adapter and it’s display mode;
- Set-up series of experiments generated pseudo-randomly or using an user-provided script file;
- Specify image exposition duration using multiple timing methods.

The software is able to load images in the following formats:

- Windows bitmap file format (BMP);
- Joint Photographics Experts Group compressed file format (JPEG);
- Truevision image file format (Targa, or TGA);
- Portable Network Graphics file format (PNG); recommended;
- DirectDraw surface file format (DDS);

¹For details about the “Curiosity Cloning – Neural Modelling for Image Analysis” project, visit the Advanced Concepts Team website: <http://www.esa.int/act/>

- Portable pixmap file format (PPM);
- Windows device-independent bitmap file format (DIB);
- High dynamic range file format (HDR);
- Portable float map file format (PFM).

1.2 Software Requirements

In order to use CCViewer, following hardware and software is required:

- PC with Microsoft Windows XP operating system (Windows Vista should work, but was not tested);
- Microsoft DirectX 9.0c runtime libraries, update August 2007 (version 9.24.1400) or newer;
- Microsoft C Runtime Library (CRT) version 7.1 or newer;
- Microsoft DirectX 9.0c-compatible video adapter.

In order to make use of the full software potential, following hardware is recommended:

- At least 2 GB of RAM (the more the better);
- High performance video adapter and display, capable of handling HD display resolutions with high refresh rates (100Hz) without interlacing, using a digital interface (like DVI-D or HDMI);
- Hard disk storage with very high transfer rates and very short access time.

1.3 Legal Information

The names of actual companies and products mentioned in this Manual may be the trademarks of their respective owners.

Initial release of this Manual has been published in 2008 by the Advanced Concepts Team, European Space Agency in the form of a technical report (number ACT-MAN-5100-CCIVUM01). If you find CCViewer software useful in experiments you conduct, please give proper credit to the CCViewer authors by quoting this technical report in related papers' references. For your convenience, we provide the appropriate BibTeX entry on listing 1.1.

Listing 1.1: This Manual's BibTeX entry

```
1 @techreport{CCIVUM01,  
2   Author={Rucinski , M.} ,  
3   Title={Curiosity Cloning Image Viewer User 's Manual} ,  
4   Institution={European Space Agency, the Advanced  
5     Concepts Team} ,  
6   Year={2008} ,  
7   Number={CCIVUM01} ,  
8   Note={Available on line at http://www.esa.int/act} ,  
9   Url={http://www.esa.int/gsp/ACT/doc/INF/pub/ACT-MAN  
    -5100-CCIVUM01.pdf}  
}
```

CCViewer software, as well as this Manual, are distributed under the following conditions:

Copyright ©2008, Advanced Concepts Team, European Space Agency All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the Advanced Concepts Team nor the name of the European Space Agency nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER

IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Chapter 2

Principles of Operation

In this chapter, important terms referring to the concepts important for the understanding of the operation of the CCViewer software are introduced, and the principles of the software operation are explained.

2.1 Terminology

Dataset – a set of image files used in an experiment;

Experiment run – a display of a sequence of a subset of images from a dataset;

Experiment – a sequence of one or more consecutive experiment runs;

Image Display Period, IDP – a period during which a single image is being displayed on the display device;

Inter-Image Period, IIP – a period between two consecutive IDPs, during which no image is displayed on the display device;

2.2 Program Operation

CCViewer is designed to conduct series of displays of sequences of images from an image dataset. When the program is run, a configuration dialog is displayed, giving the user the opportunity to set up the experiment and different image presentation parameters. Program configuration is stored in a file named “ccviewer.ini” located in the same directory as the program executable. This allows preserving program configuration from one program run to another. The configuration file is written every time the user presses the “OK” button in the configuration dialog.

After the experiment is set up, pressing the “OK” button in the configuration dialog starts the experiment. An experiment consists of at least one

display of a sequence of images. Displayed sequences may be different, but they may use only images available in the dataset.

Every image sequence display consists of the following parts:

1. Pre-loading of images from a mass storage to the system memory in order to improve performance. Because it may take considerable time, during this phase a progress bar is displayed to the user, allowing monitoring the progress and preparation to the experiment run itself.
2. Eye fixation screen. This is a blank screen with neutral background with a fixation cross displayed in the centre of the screen, allowing the user to fully concentrate right before the experiment run. In addition to that, a countdown counter is displayed, allowing the user to precisely anticipate the actual experiment run start what is supposed to reduce the surprise effect. Duration of the fixation screen is configurable.
3. Actual experiment run. The images are displayed to the user in appropriate sequence. Optionally, images may be interleaved with Inter-image Periods. Duration of an image exposition and IIPs may be configured independently. Also, a semi-transparent fixation cross may be kept on the screen during this phase.
4. Eye rest screen. After the last image is presented, a blank screen with neutral background is presented to the user for the same duration as the fixation screen in order to reduce the surprise effect at the experiment run end. No fixation cross nor countdown is displayed.
5. If there is more than one experiment run, the experiment continues from the point 1.

Chapter 3

Configuration Dialog

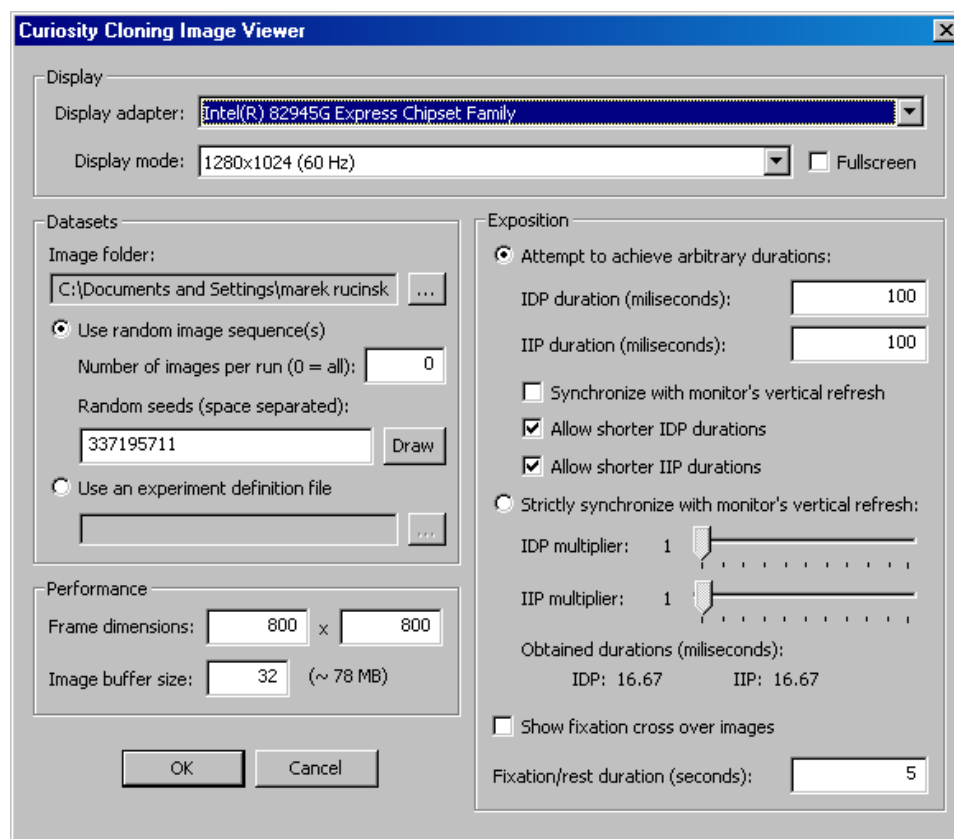


Figure 3.1: CCViewer configuration dialog

Configuration dialog of the CCViewer is shown on figure 3.1. All dialog controls and their meanings are described below.

3.1 Group “Display”

This group contains controls related to the display adapter.

3.1.1 “Display adapter”

This control allows the user to select the display adapter which will be used to display the images. This option is useful in computer systems in which a separate graphics adapter is used to handle the high-definition display or when different interfaces of the graphics adapter are handled as if they were separate adapters (this approach has been commonly used by ATI).

3.1.2 “Display mode”

This control allows the user to select the video mode to be used to display images. For guidelines on video mode selection, please refer to the section 4.1.

3.1.3 “Fullscreen”

This checkbox toggles between full-screen and windowed program operation. Windowed mode is made available mainly for software developing purposes. During normal program operation, full-screen mode should be used, as it makes use of entire screen (no other program windows are visible) and may very likely provide better performance.

3.2 Group “Datasets”

This group contains controls related to the image datasets.

3.2.1 “Image folder”

This control allows the user to point to the directory which contains the images to be displayed. Folder picking dialog is displayed using the “...” button.

3.2.2 “Use random image sequence(s)” / “Use an experiment definition file”

These two radio buttons allow the user to choose between the two options for defining an experiment scenario, i.e. the image sequence(s) to be displayed. See section 4.2 for more information on defining experiment scenarios.

3.2.3 “Number of images per run”

In this edit box, the user can specify how many images should be contained in one experiment run. This number must be less than or equal to the number of images available in the dataset folder. The value “0” means that all images in the image folder should be used.

3.2.4 “Random seeds”

This edit box allows the user to specify random seeds which will be used to generate image sequences for every experiment run. In consequence, the number of entered random seeds determines the number of experiment runs. If there is more than one experiment run, consecutive random seeds should be separated by a space. Random seeds may be entered manually or generated pseudo-randomly, using the “Draw” button.

3.2.5 Experiment definition file (not labelled)

When using an experiment definition file to define the image sequences to be displayed, the “...” button allows the user to point to the file to be used. Format of this file is described in section 4.2.

3.3 Group “Exposition”

3.3.1 “Attempt to achieve arbitrary durations” / “Strictly synchronize with monitor’s vertical refresh”

This pair of radio buttons allows the user to choose between available timing options. For more information about timing options and differences between them, see section 4.3.

3.3.2 “IDP duration”

This edit box allows the user to enter the desired duration of the Image Display Period, in milliseconds.

3.3.3 “IIP duration”

This edit box allows the user to enter the desired duration of the Inter-Image Period, in milliseconds. The value “0” will cause that the Inter-Image Period will be completely omitted.

3.3.4 “Synchronize with monitor’s vertical refresh”

This checkbox toggles the synchronisation with the vertical refresh of the display device.

3.3.5 “Allow shorter IDP durations”

This checkbox toggles the frame display time correction during the Image Display Period. Please refer to the section 4.3 for details.

3.3.6 “Allow shorter IIP durations”

This checkbox toggles the frame display time correction during the Inter-Image Period. Please refer to the section 4.3 for details.

3.3.7 “IDP multiplier”

This track bar allows the user to define the duration of the Image Display Period as a multiplicity of the vertical refresh period of the display device. Resulting IDP duration in milliseconds is displayed on an appropriate caption below the track bars.

3.3.8 “IIP multiplier”

This track bar allows the user to define the duration of the Inter-Image Period as a multiplicity of the vertical refresh period of the display device. Resulting IIP duration in milliseconds is displayed on an appropriate caption below the track bars. The multiplier of “0” completely disables the Inter-Image Period.

3.3.9 “Show fixation cross over images”

This checkbox toggles display of the fixation cross over the images during the experiment. The cross is partially transparent.

3.3.10 “Fixation / rest duration”

This edit box enables the user to enter the desired duration of both eye-fixation screen (a blank screen with a fixation cross and countdown, shown before the actual experiment run) and the eye-rest screen (a blank screen shown after the experiment run is finished). The time is specified in seconds.

3.4 Group “Performance”

This group contains controls which allow the user to tune the performance of the application. For detailed information regarding the application’s performance, please refer to the section 4.4.

3.4.1 “Frame dimensions”

These two edit boxes allow the user to specify the dimensions (respectively, width and height, in pixels) of the images contained in the dataset. Actual size of the images may be different from specified, but please note that bigger images will be cropped to the frame size (left-top part of an image is cut), and smaller images will be placed in the left-top part of the frame.

3.4.2 “Image buffer size”

In this edit box, the user may specify the number of images to be pre-loaded to the system memory before starting the actual image display. Approximation of the amount of the memory required (in megabytes) is displayed on a caption next to the edit box. Please note that both frame dimensions and the number of images affect the amount of memory required.

Chapter 4

Usage Guidelines

4.1 Video Mode Selection

There are various factors affecting the optimal selection of the display mode. The first one is the size of the images displayed during the experiment. Ideally, width and height of the screen in selected display mode should be identical to the dimensions of images being displayed. However, it is very often not possible. In such case, the best video mode would be the one with the smallest dimensions which guarantee that entire images will be displayed. In such case, the images will be rendered in the centre of the screen.

For example, if the dataset consists of images of size 800 by 800 pixels, and the monitor supports display modes with resolutions 800 by 600, 1024 by 768 and 1280 by 1024, the latter should be chosen, as it is the only mode with height greater than or equal to 800. The user may decide however to use a display mode which is smaller than image dimensions. In such case, the central part of the image will be displayed.

The second very important factor affecting the video mode selection is the mode's refresh rate. Refresh rate defines how many times per second the screen is completely redrawn. This parameter is important, because it has a big impact on display timing. The higher the refresh rate, the more flexibility the user has when defining timing parameters while keeping the high quality of the displayed content.

Last but not least, optimal display mode selection is different for different types of display devices. One of the major practical differences between CRT and LCD monitors is that the former display high quality image regardless of the video mode being selected. That means, that the user can freely select the video mode basing on the criteria mentioned above (appropriate dimensions with as much refresh rate as possible). In contrary, LCD devices obtain optimal display quality only when working in the display mode which is native for the device. Usually this is the highest display mode (i.e. the

one with biggest screen dimensions) supported by the device. Support for the other modes is in most cases only emulated, i.e. the graphics hardware automatically resizes rendered image to the native size of the monitor, what has a very significant impact on the image quality (usually heavy blur). Thus, on LCD devices the best choice most likely will be to use the native resolution, regardless of the size of displayed images.

4.2 Setting-up the Experiment

There are two ways in which the user may define the experiment, i.e. the sequences of the images to be displayed to the subject. The first method is to generate the image sequences using a pseudo-random number generator. The second one is to provide an experiment definition file.

The pseudo-random method uses the following algorithm to generate the image sequences to be used in the experiment:

1. For every provided random seed:
 - (a) Take all images in the dataset folder in the order determined by the lexicographic order of their file names
 - (b) Generate a random permutation of the files basing on the current random seed
 - (c) Take first N images from the sequence (where N is a configuration parameter, being the number of images per experiment run) or the whole sequence, if N is equal to 0

Image sequences generated in this way have the following properties:

- The number of sequences (experiment runs) is equal to the number of random seeds provided by the user;
- During each experiment run, every image is displayed not more than once;
- One image may appear in multiple runs;
- Sequences of images are independent from each other, as the input permutation is always the same (determined by the lexicographical order of the file names).

Note, that due to the nature of the pseudo-random number generators, generated image sequences may be considered random, but are completely determined by the random seeds used. Thus, in order to repeat exactly the same experiment, one just has to use identical random seeds.

Please also note that the program has no knowledge about the paradigms of experiments involving the display of visual stimuli — it just displays the

sequences of images. It does not discern between background, oddball and distraction images. This makes the random sequence generator rather of small use in for instance oddball paradigm experiments.

The second method of defining the experiment is to provide an experiment definition file. This file contains the descriptions of sequences of images to be displayed.

The format of the file is the following. The file should be a plain ASCII text file. The first line of the file should contain the number of file sequences defined in the file (i.e. the number of experiment runs). Following should be the appropriate number of file sequence definitions, separated by one blank line. Each file sequence definition consists of names of files making up the sequence, one per line. File names are relative to the image folder directory. Before launching the experiment, the program verifies if all the files referred to in the experiment definition file are actually present in the image dataset folder.

Example of an experiment definition file can be faound on listing 4.1

Listing 4.1: Example experiment definition file

```

1  3
2  File1.png
3  File2.png
4  File3.png
5  File4.png
6
7  File5.png
8  File6.png
9  File5.png
10
11 File4.png
12 File2.png

```

Experiment definition files have the following advantages over pseudo-randomly generated sequences:

- The user has the full control over the sequences *a priori*;
- Experiment runs can be of different length;
- One image may appear multiple times in one experiment run;
- Experiment definitions may be generated using external tools, for instance designed for modelling the oddball paradigm.

4.3 Timing Options

There are two basic timing strategies available for the user of the CCViewer. They offer different levels of flexibility, accuracy and presentation quality.

When selecting an appropriate timing method, a trade-off must be done between these three properties.

4.3.1 Timing Basics

The basic mechanism behind image display timing in CCViewer is the frame scheduler. All frames – any contents of the screen, whether it is an image, a fixation screen or a blank screen – before they are displayed, are assigned a time moment at which they should be displayed. The program runs in a loop, checking the value of a high-precision system timer. When current system time matches or exceeds due time of the next frame, the frame is rendered to the screen. After this is done, next frame to be rendered is scheduled. This algorithm is shown on figure 4.1.

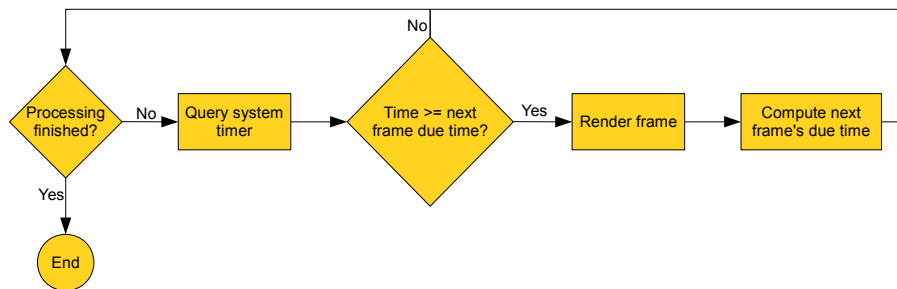


Figure 4.1: Frame rendering algorithm

Using the high-performance timer mechanism (*QueryPerformanceFrequency* and *QueryPerformanceCounter* Windows API functions) to measure the time pass is the most accurate timing mechanism available in the Microsoft Windows XP operating system. Other available timing mechanisms, like for instance Queue Timers, do not provide satisfying accuracy.

This timing solution is not perfect though. There exists unavoidable and unpredictable difference between frame due time and the time at which the frame is actually displayed. Firstly, the program queries the timer with a finite resolution, usually significantly lower than the timer resolution. In consequence, the program usually detects, that the scheduled frame time *has already passed* rather than that the *time is now*. Secondly, after the condition is detected, rendering the frame also takes some unpredictable amount of time. Thus, the outcome of using the basic timing mechanism may be visualised as on figure 4.2.

As shown on the figure, actual frame display time succeeds the scheduled frame time by a random, unpredictable amount of time (indicated by red rectangles on the time axis). Timing strategies mentioned in the beginning of this section differ in the way the next frame due time is calculated

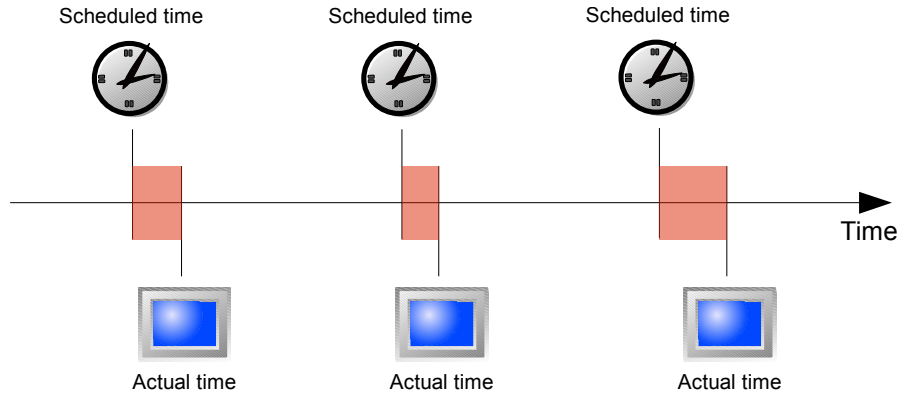


Figure 4.2: The basic timing mechanism

and in the method the frame display is triggered. Details of these strategies are described in the following sections.

4.3.2 Arbitrary Timing

The first method assumes that the user specifies an arbitrary duration of a frame display time¹. The program tries to meet user's expectations as closely as possible.

In the basic version of this method, the due time of the next frame is calculated relatively to the actual display time of the previous frame. If d denotes the desired frame duration, program operation may be visualised as on figure 4.3.

Because of the unpredictable delay between scheduled and actual frame display time, resulting time intervals (on the figure denoted as d' and d'') between actual frame display times are always different from and not shorter than the desired interval d . The advantage of this approach is that for every frame it is guaranteed that it will be displayed for at least d time units. The drawback is that the average frame presentation rate is always lower than expected, i.e. lower than the rate calculated basing on arbitrary frame duration given by the user.

If obtaining accurate presentation rate is more important than achieving a guaranteed exposition time, one may try to compensate the error introduced by the random delay by changing the reference point from which the due time of the next frame is calculated. In order to achieve expected frame presentation *rate*, the due time of the next frame must be calculated relatively to the *due time* of the previous frame, as illustrated on figure 4.4.

¹Please keep in mind that the term “frame” means here any contents of the screen, including a blank screen, so there is no need to discern between IDP and IIP.

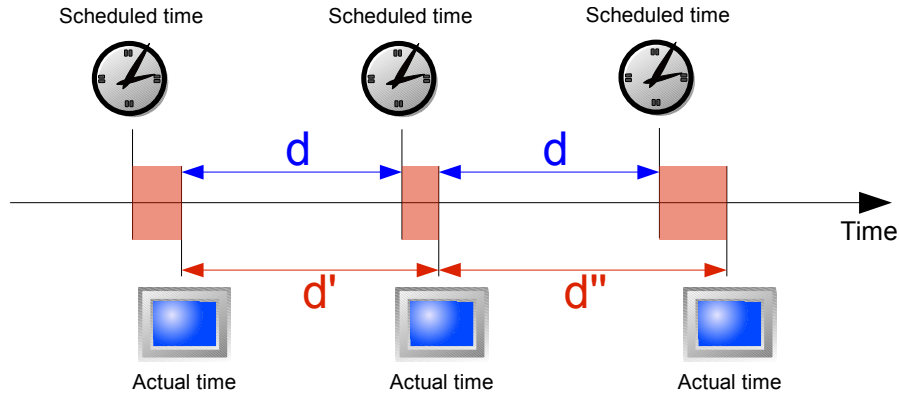


Figure 4.3: Outcome of the arbitrary timing approach

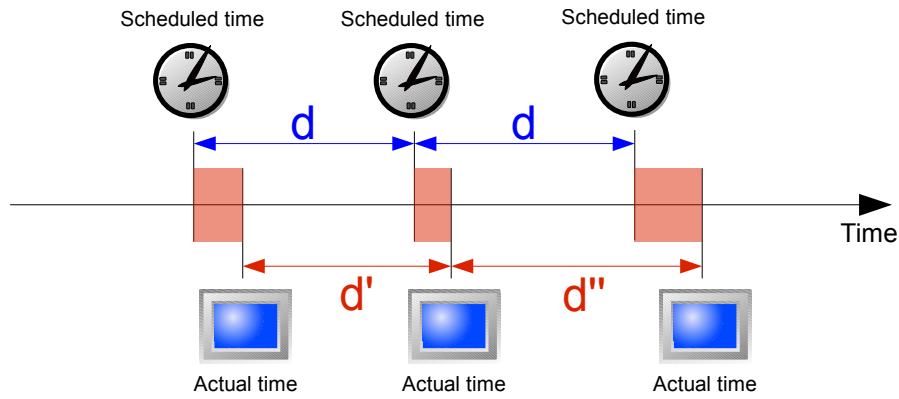


Figure 4.4: Due-time relative arbitrary timing approach

Although actual frame exposition times d' and d'' are still different from d , they now oscillate around its value. In other words, some frames are displayed for a time longer than d , and some for shorter, hopefully with a symmetrical probability distribution.

In the CCViewer GUI, frame display delay compensation is enabled using two options in the “Exposition” controls group, namely “Allow shorter IDP” and “Allow shorter IIP” checkboxes. These options allow selection of the due time calculation method for IDP and IIP independently, thus allowing for instance forcing the display of the images to last for the desired value, while allowing shortening the duration of the blank screen between two images in order to compensate the drift of the average image presentation rate.

Practical difference between the two presented timing options is illustrated on the histograms on figure 4.5. Data has been gathered for a sample

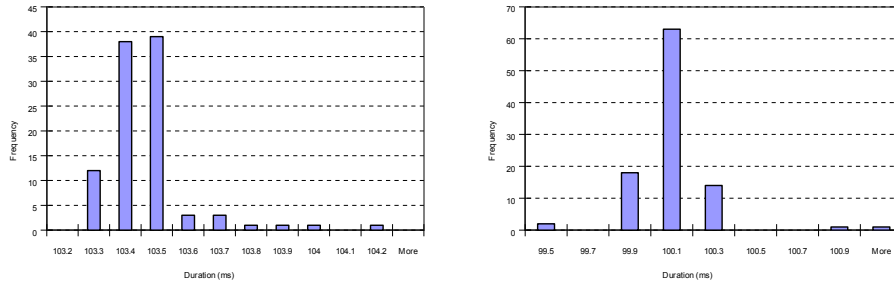


Figure 4.5: Timing histograms for arbitrary timing without (left) and with lag compensation (right)

consisting of 100 images displayed with desired exposition interval of 100 milliseconds. The left-hand histogram shows the distribution of the actual measured frame display time when the due time of the next frame is calculated relatively to the previous frame’s actual display time (the basic version); the right hand histogram – when it is calculated relatively to the previous frame’s due time (the second version).

It is clear that for the former method, no frame is displayed for a time shorter than desired 100ms. For the latter, the distribution is concentrated around the desired value of 100ms (most measurements fall between 99.9ms and 100.1ms).

Another very important practical issue related to the arbitrary timing method is synchronisation of the frame display with so-called vertical refresh period of the display device. For CRT monitors, refresh rate is directly related to the trajectory of the electron beam inside the kinescope. For LCD displays, one of the most important parameters determining the device’s refresh rate is the rate of on-off pulses of the monitor’s backlight.

When the software displays an image on the device without taking into account it’s refresh rate, it is very likely that the screen contents will change while the device is actually drawing previous screen contents. The result will be an image consisting of a part of the previous constants in the upper part, and the new contents in the lower part. This effect is often referred to as “tearing” or “flickering” and affects both CRT and LCD displays. It is unfortunately both very easily noticeable and quite disturbing for the spectator. In order to increase the quality of the exposition, the update of the screen contents must be allowed only during so-called *vertical refresh period*, i.e. during the time period when no contents are actually being drawn on the display device (in CRT monitors this is the time when the electron beam travels from the bottom-right corner to the upper-left corner of the screen). This is usually achieved at the hardware level by delaying the actual frame display until the next vertical refresh period. The outcome of synchronising the image display with the vertical refresh period of the display

device is illustrated on figure 4.6.

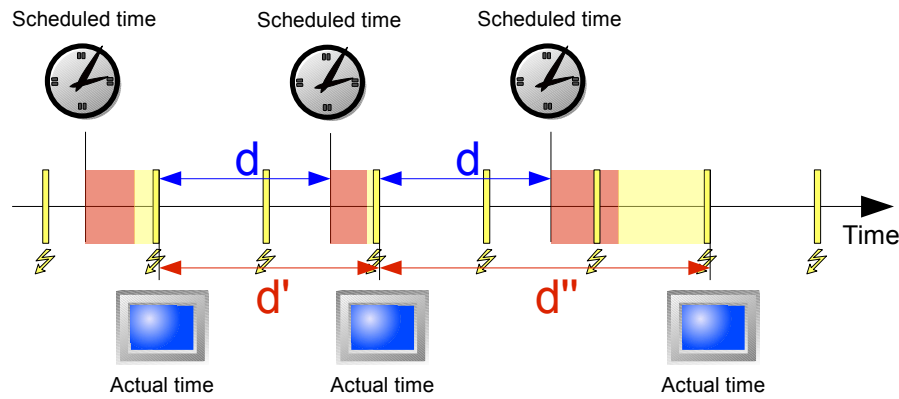


Figure 4.6: Outcome of the arbitrary timing method with vertical refresh synchronisation

On the illustration, vertical refresh periods of the display device have been depicted as vertical yellow bars. As clearly seen, synchronisation of an arbitrary frame duration with the vertical refresh period of the display device may have a dramatic impact on the actual frame exposition time, which is now forced to be a multiplicity of the length of the interval between two consecutive vertical refresh periods of the display device. In consequence, the delay between frame's due time and actual display time is further increased by the time from the moment when the frame has been rendered to the next vertical refresh period (depicted as yellow rectangles on the time axis). Enabling lag compensation does not always improve the situation because the delay caused by the processing time is random. Depending on the relation between the desired frame exposition duration d and the refresh rate of the display device, resulting actual frame exposition time may be significantly different for different frames. The example resulting distributions of the frame exposition times are illustrated on histograms on figure 4.7. Frame display was in this case synchronised with the vertical refresh rate of an LCD monitor (60 Hz, meaning roughly 16.67 milliseconds between two vertical refresh periods). Again, on left hand histogram the compensation is disabled, on the right hand – enabled.

It is clear that apart from increasing the image presentation quality (“tearing” has been eliminated), one has gained significant improvement of the frame exposition time variance. This is the consequence of eliminating (at least partially) the delay resulting from the random frame rendering time. The drawback is that the actual frame exposition time may very well be significantly different from user's demands. It may also happen that if the relation between desired exposition time and device's refresh rate

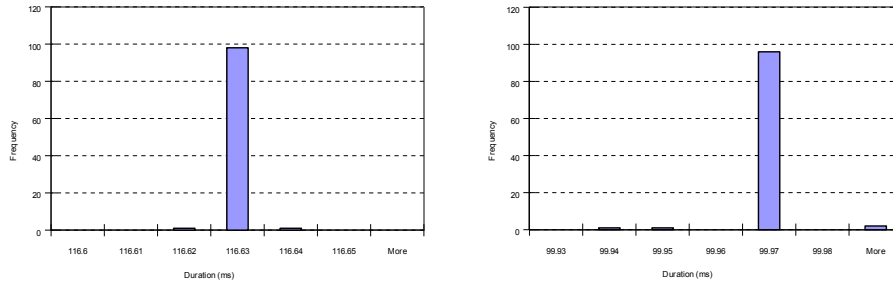


Figure 4.7: Timing histograms for synchronised timing without (left) and with lag compensation (right)

is inconvenient, resulting distribution will be multimodal. All this is the consequence of synchronising to the fixed refresh rate – it implies that the exposition time may be no longer selected arbitrarily. This observation is the fundamental idea behind the second timing method available in CCViewer.

4.3.3 Synchronised Timing

As indicated in the previous section, if the image presentation quality must be kept at the highest level, frame display must be synchronised with the refresh rate of the display device. This implies that the frame exposition time may be selected only as a certain multiplicity of the time period between two consecutive vertical refresh periods of the display device. Thus, in the second timing method the frame exposition time is no longer specified directly by the user, but calculated as a pre-defined number of such intervals.

Obviously, the higher the refresh rate, the shorter the interval between vertical refresh periods, and the more flexibility the user has when selecting the exposition time. Table 4.1 shows example exposition times obtainable for most common refresh rates of the display devices available in the consumer market. As clearly seen, even for the high-end refresh rates, the flexibility of the frame duration selection is significantly reduced in comparison to the arbitrary method. However, if available display device supports multiple refresh rates, one can choose among much more options.

Special care has been taken by the CCViewer developers to ensure as reliable actual frame exposition duration as possible. Unfortunately, Microsoft DirectX 9 programming interface used by CCViewer does not allow the software to reliably count the number of vertical refresh periods that have passed during certain period of time. Thus “Skipping” the desired number of vertical refresh periods has to be obtained by calculating the duration of the interval between vertical refresh periods basing on the known refresh rate, and scheduling the next frame shortly after one-but-last vertical refresh period. This allows the software to have enough time to render the frame

Vblanks per frame	Monitor Refresh Rate (Hz)					
	50	60	70	75	85	100
1	20.00	16.67	14.29	13.33	11.76	10.00
2	40.00	33.33	28.57	26.67	23.53	20.00
3	60.00	50.00	42.86	40.00	35.29	30.00
4	80.00	66.67	57.14	53.33	47.06	40.00
5	100.00	83.33	71.43	66.67	58.82	50.00
6	120.00	100.00	85.71	80.00	70.59	60.00
7	140.00	116.67	100.00	93.33	82.35	70.00
8	160.00	133.33	114.29	106.67	94.12	80.00
9	180.00	150.00	128.57	120.00	105.88	90.00
10	200.00	166.67	142.86	133.33	117.65	100.00
11	220.00	183.33	157.14	146.67	129.41	110.00
12	240.00	200.00	171.43	160.00	141.18	120.00

Table 4.1: Exposition times for popular refresh rates (in milliseconds)

before the target vertical refresh period comes. However, as experiments with the software on different hardware platforms showed, that the actual refresh rate of a display device may be somewhat different from the nominal one. If the actual refresh rate is a bit lower than the nominal one, it may happen that the frame will be displayed too early (during the one-but-last vertical refresh period). Because no official tolerance specification standards have been found, CCViewer authors introduced a margin of around 16.6% of the nominal between-refresh period, which turned out to be suitable for all tested devices. Finally, program operation in the synchronised timing mode is illustrated on figure 4.8.

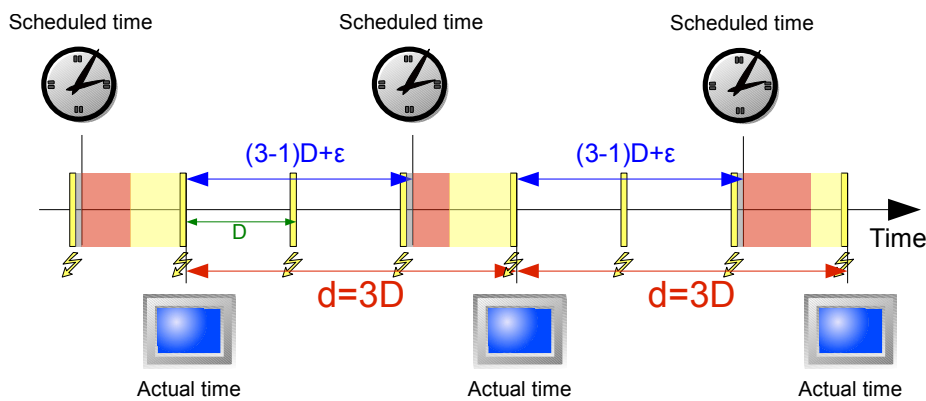


Figure 4.8: Outcome of the synchronised timing method

On the figure, D denotes the duration of the time interval between ver-

tical refresh periods, 3 is the desired duration of the frame exposition time expressed in the number of intervals, and ϵ denotes the assumed safety margin. It is clearly visible, that if the worst-case rendering time of a frame is small enough to fit between two vertical refresh periods, the resulting actual timing is very precise (equal to desired $3 \cdot D$).

4.3.4 Final Remarks

In this subsection, several guidelines for timing method selection have been gathered.

- Rather avoid using the arbitrary timing method. The only advantage – flexibility – comes at the cost of the presentation quality. Synchronising with the refresh rate of the display device removes the only advantage over the synchronised mode, while the latter is much more reliable;
- In order to increase the flexibility, use display devices which support multiple refresh rates (*not* by emulation);
- Keep in mind that using devices with very high refresh rates requires more robust display hardware. This is because the software is given less time to render the frame between consecutive vertical refresh periods. If desired exposition interval is obtainable using multiple refresh rates, choose display mode with the lower refresh rate – it will be more reliable;
- Make sure, that the display hardware does not make any tricks with the refresh rate. It has been verified by the program authors, that some display hardware when using multiple displays applies different refresh rates for both display devices *and* the software. The result was that actual exposition intervals were completely different from what was calculated basing on the refresh rate of selected display modes. The conclusion of the experiments is to avoid using multiple displays at once. If an auxiliary video output has to be used (for instance when using a laptop and a projector), disable the main display.

4.4 Performance Considerations

Special care has been taken that the CCViewer operates as reliably as possible. Nevertheless, the user must also keep in mind that certain things affect the software's performance.

Most crucial parameter for achieving high performance of the CCViewer is the amount of available system memory. In order to display an image, it has to be transferred from a mass storage (usually a hard disk) to the system

memory and then to the graphics adapter memory. The throughput of the mass storage is usually not sufficient for displaying images with very fast rate directly from it. In order to achieve high image display rates, CCViewer allows pre-loading of a certain number of images to the system memory before starting the display. The highest reliability is of course obtained when all images to be displayed are pre-loaded (slow hard disk access is completely eliminated during the image display phase). The number of pre-loaded images is set in the CCViewer GUI using the “Image buffer size” edit box in the “Performance” control group.

Program memory requirements are directly proportional to the number of pre-loaded images and to the size of the images themselves. Images have to be kept in the computer memory in an uncompressed bitmap format, thus the amount of memory required for an image grows rather fast with the image dimensions. For example, 2 gigabytes of available memory would allow storing only a bit less than 540 images of dimensions 1000 by 1000 pixels. On 32-bit processors, Microsoft Windows operating system does not allow any program to use more than 2 gigabytes of system memory. Moreover, it is very unlikely that such amount of memory will be actually available for the program. Even if the program is allowed to run, part of it’s memory will be swapped (i.e. put to the hard disk), neutralising the performance gain achieved with the buffer.

CCViewer allows the user to specify what is the size of images being displayed (using “Frame dimensions” edit boxes in the “Performance” control group). The given dimensions do not have to match actual dimensions of displayed images. If they are smaller, they will be loaded in the upper-left part of the frame buffer. If they are bigger, the left-upper part of the image will be “cropped” and displayed. For the user’s convenience, in the “Performance” control group an estimate of required memory amount is displayed and updated in real-time when either image dimensions or buffer size is changed.

Summarising, in order to obtain maximum performance:

- Use system with as much random access memory as possible (but note that 32-bit version of Microsoft Windows XP supports maximum 3 gigabytes);
- Specify the image buffer size equal to the number of images displayed;
- Set the image size to be exactly equal to the size of the images being displayed (not bigger);
- If the required amount of memory is too high, specify largest possible buffer size, and use the fastest disk storage available (possibly even a RAID matrix);

- Make sure no other software is running in the background during the experiment. Especially, disable any anti-virus software if present, as it may very likely scan all files loaded by the CCViewer against viruses;
- Make sure to disable any system services which are not necessary, as they consume system memory;

4.5 Logging

In order to trace the conducted experiments CCViewer creates a log file in which much information about the program usage is stored. The log file is created in the same directory as the CCViewer's executable file. The log file is a plain ASCII text file structured in the following way:

1. A header containing complete experiment configuration;
2. For every experiment run:
 - (a) The sequence of the file names of the images being displayed;
 - (b) Start time of the image display (in human-readable form);
 - (c) Image display timing information, that is:
 - i. Frequency of the high-performance system timer;
 - ii. Basic statistics for IDP and IIP measurements;
 - iii. Full table of measurements for every displayed image for both IIP and IDP in both raw timer units and milliseconds in both cumulative and non-cumulative form;
 - (d) Basic statistics for image loading times;

“Basic statistics” include the sample count, mean and standard deviation.